Amendments to the Claims:

Claims 1-27 are pending. This listing of claims will replace all prior versions and listings of claims in the application:

Listing of Claims:

1. (currently amended) A client-server system capable of validating cached eXtensible Markup Language (XML) data comprising:

a data store for storing XML data;

a server for retrieving and updating XML data in the data store to service client requests;

a transformation engine for transforming XML data into a format suitable for a client application based on a set of transformation rules;

a cache for temporarily storing transformed XML data as data objects for later reuse;

a cache monitor for ensuring that cached objects are validated when changes to XML data in the data store are detected by the server; and

an object dependency mapper for automatically and continuously determining dependencies between XML data in the data store and sets of transformation rules.

2. (original) The system as recited in claim 1 further comprising an object manager for managing data objects in the cache.

3. (original) The system as recited in claim 2 further comprising a transformation rule alert service for detecting when the transformation rules are modified, added to the system and deleted from the system.

4. (original) The system as recited in claim 3, wherein the server accesses the object manager to generate a response to a client request for data.

5. (original) The system as recited in claim 4, wherein the server accesses the cache monitor to validate cached objects when a data update request is received.

6. (original) The system as recited in claim 5, wherein:

data in the data store is represented as a tree structure having a root node, a plurality of intermediate nodes and leaf nodes, the leaf nodes representing data in the data store; and

a transformation rule is an expression describing a path from the root node to a particular node in the tree.

7. (original) The system as recited in claim 6, wherein:

a set of the transformation rules constitutes a style sheet; and

the transformation engine receives a style sheet and the data tree as input, and outputs a transformed data object.

8. (original) The system as recited in claim 7, wherein the cache includes a plurality of data objects each associated with a style sheet used to generate said each object.

9. (original) The system as recited in claim 8, wherein:

the object manager uses the transformation engine to generate a new object in response to a client request when the new object does not exist in the cache; and

the object manager stores the new object in the cache automatically.

10. (original) The system as recited in claim 9, wherein the object manager periodically refreshes the cache and removes the objects that have been flagged as invalid by the cache monitor.

11. (original) The system as recited in claim 9, wherein the object manager optionally maintains statistical information for each object in the cache, and automatically removes cached objects that are being accessed infrequently by the clients.

12. (original) The system as recited in claim 7, wherein the object dependency mapper includes a table of dependencies, each dependency associating a transformation rule with the style sheets that include the transformation rule.

13. (original) The system as recited in claim 12, wherein the table of dependencies is automatically generated and maintained by the object dependency mapper.

14. (original) The system as recited in claim 12, wherein the transformation rule alert service communicates updates on the style sheets to the object dependency mapper.

15. (original) The system as recited in claim 12, wherein the cache monitor uses the table of dependencies to determine a set of relevant style sheets, said relevant style sheets referencing a node in the data tree related to a data update; and

the cache monitor accesses the cache to invalidate the objects generated by the relevant style sheets.

16. (currently amended) In a client-sever computing system having a cache and storing eXtensible Markup Language (XML) data as data objects, a method for determining invalid cached objects comprising ~~the steps of~~:

transforming XML data into a format suitable for a client application based on a set of transformation rules;

determining dependencies between cached objects and XML data related to the cached objects;

monitoring updates to the related XML data; and

determining the cached objects that are affected by changes to the related XML data based on the dependencies.

17. (original) The method as recited in claim 16, wherein the transformed format is html.

18. (original) The method as recited in claim 16, wherein:

a set of transformation rules constitutes a style sheet;

each dependency associates a transformation rule with a style sheet; and

each data object is the data transformed by a style sheet.

19. (original) The method as recited in claim 16, wherein:

data is represented as a tree structure having a plurality of nodes; and

the cached objects that are affected by the data changes are determined using the tree structure.

20. (original) The method as recited in claim 19, wherein the dependencies are maintained in a table of dependencies.

21. (currently amended) The method as recited in claim 20, wherein the step of determining the affected objects comprises ~~the steps of~~:

identifying the nodes associated with data updates;

identifying the transformation rules related to the identified nodes;

determining a set of relevant style sheets using the table of dependencies, the relevant style sheets including the identified transformation rules; and

identifying the cached objects that have been transformed by the relevant style sheets.

22. (currently amended) In a client-sever computing system having a cache and storing eXtensible Markup Language (XML) data as data objects, a computer program product for determining invalid cached objects comprising:

a computer readable medium;

means, provided on the computer readable medium, for transforming XML data into a format suitable for a client application based on a set of transformation rules;

means, provided on the computer readable medium, for determining dependencies between cached objects and XML data related to the cached objects; ~~and~~

means, provided on the computer readable medium, for monitoring updates to the related XML data; and

means, provided on the computer readable medium, for determining the cached objects that are affected by changes to the related XML data based on the dependencies.

23. (original) The computer program product as recited in claim 22, wherein the transformed format is html.

24. (original) The computer program product as recited in claim 22, wherein:

a set of transformation rules constitutes a style sheet;

each dependency associates a transformation rule with a style sheet; and

each data object is the data transformed by a style sheet.

25. (original) The computer program product as recited in claim 22, wherein:

data is represented as a tree structure having a plurality of nodes; and

the cached objects that are affected by the data changes are determined using the tree structure.

26. (original) The method as recited in claim 25, wherein the dependencies are maintained in a table of dependencies.

27. (original) The computer program product as recited in claim 26, wherein the means for determining the affected cached objects comprises:

means, provided on the computer readable medium, for identifying the nodes associated with data updates;

means, provided on the computer readable medium, for identifying the transformation rules related to the identified nodes;

means, provided on the computer readable medium, for determining a set of relevant style sheets using the table of dependencies, the relevant style sheets including the identified transformation rules; and

means, provided on the computer readable medium, for identifying the cached objects that have been transformed by the relevant style sheets.